IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON NEW WAVEFORM DESIGN AND AIR-INTERFACE FOR FUTURE HETEROGENEOUS NETWORK TOWARDS 5G

Received October 4, 2018, accepted October 21, 2018, date of publication October 24, 2018,
date of current version November 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2877767

An Evolutionary Scatter Search Particle Swarm Optimization Algorithm for the Vehicle Routing Problem With Time Windows

JINGTIAN ZHANG, FUXING YANG, AND XUN WENG
Automation School, Beijing University of Posts and Telecommunications, Beijing 100876, China
Corresponding author: Jingtian Zhang (buptzjt@163.com)

ABSTRACT Vehicle routing problem with time windows (VRPTW) contains two crucial objectives: minimizing the number of vehicles and minimizing the total travel distance. However, most algorithms focus on the number of vehicles, while the travel distance should be considered as the primary objective in some practical situations, especially in the modern logistics. Research has shown that designing a systematic framework to combine multiple algorithms with different characteristics will significantly improve the overall performance of the hybrid algorithm. This paper proposes an evolutionary scatter search particle swarm optimization algorithm (ESS-PSO) to solve the VRPTW with the objective of minimizing the total travel distance. In ESS, a genetic algorithm and a new "route+/−" evolutionary operator are introduced in scatter search template. In addition, we proposed a discrete PSO that sets the route-segment as the velocity of particles and in which the velocity and position updating rules are designed based on the concept of "ruin and recreate." These two algorithms work in a cascade learning architecture, in which PSO learns from the exemplary solutions in the reference set maintained by ESS. The search direction of the algorithm is adjusted by analyzing the relationship between the number of vehicles and the total travel distance in real time. We designed a new solution representation called "auxiliary code" based on customer allocation to maintain the diversity of the reference set. Experiments with the Solomon benchmark show that ESS-PSO is effective and efficient, and it achieves very competitive results, especially in the datasets of the category "2."

INDEX TERMS Genetic algorithm, particle swarm optimization, optimal scheduling, scatter search, vehicle routing problems with time windows.

I. INTRODUCTION

The Vehicle Routing Problem with Time Windows (VRPTW) is a logistics and distribution management problem which has many real-world applications such as supply chain management and express delivery [1]. It is one of the most important and difficult NP-hard combinatorial optimization problems [2] and has been selected as a classic test problem for performance verification by many algorithms. Solomon benchmark [3] is the most well-known test dataset in VRPTW which has been widely studied.

The exact algorithms for solving VRPTW can obtain optimal solutions [4], but the performance of the algorithms mainly depends on the structure of the problem and the computing time grows exponentially when dealing with large-scale issues. The latest exact algorithm [5] has been able to obtain all the optimal solutions of 56 examples of Solomon benchmark(100 customers), but the computational time in the dataset of category "2" with a wide time window is still unsatisfactory (e.g. the average computational time on Intel X-ES2637 3.5GHz CPU of R2 instances is 6432 seconds in which instance R208 is solved in 64105 seconds). The meta-heuristic algorithms [6] can obtain a "good enough" solution in a short time and have the capacity to solve the large-scale complex problems, which is more suitable for application in practical situations.

In meta-heuristics, the algorithms based on the concept of "Memories and Guidance" have achieved excellent results. Rochat and Taillard [11] first proposed the Adaptive Memory (AM) which stores good solutions or partial solutions. The new solution is created by combining the promising solution components (routes) in the AM. Based on AM, many modified methods achieved good results [12], [14]. Russell and Chiang [1] used the scatter search algorithm to solve VRPTW and pointed out that the method

proposed by Rochat and Taillard [11] is very similar to the scatter search algorithm where AM is equivalent to the ''reference set'' according to the framework of algorithms. In some population-based algorithms, Baños *et al*. [15] and Wu *et al*. [16] used an external archive to store the non-dominated solutions. Barbucha [18] proposed a guiding framework for the parallel cooperative search using a so-called ''solution warehouse'' to store the good solution. It is noteworthy that all of the above algorithms have used various means to maintain the diversity of the population to avoid premature convergence. Balancing the quality and diversity of the elite populations is critical to the ''Memories and Guidance'' algorithms.

With the strong constraints, VRPTW does not have a clear neighborhood structure. It means that feasible solutions may not locate in the neighborhood of any candidate solutions in the searching space [19]. ''Ruin and recreate'' approach partially destroys the current solutions and reinsert the removed part into the ''destroyed'' partial solution by using some heuristics, and it is always able to obtain feasible solutions. There are many researchers used ''Ruin and recreate'' approach for solving VRPTW [20]–[22]. For this kind of ''construct'' algorithm, the introduction of randomness outside ''greedy rules'' can significantly improve the effectiveness and efficiency [24].

As a classic heuristic, PSO has attracted great attention of researchers due to its attributes of high efficiency, fast optimization speed, and simple implementation. However, since it is more suitable for solving continuous optimization problems, the successful applications of PSO on VRPTW is still less. Some researchers use conventional PSO to solve this problem by encoding the solutions into real numbers [29], [30]. The solution representation is always too long in this way which would limit the performance of solving large-scale problems. Gong *et al*. [31] proposed a set-based PSO taking the searching space as a set of arcs and transforming the PSO into discrete forms. The experimental results show that the discrete PSO algorithm is very effective for VRPTW. Wu *et al*. [16] introduced a multi-objective algorithm based on the same discrete PSO as Gong *et al*. PSO has strong local searching ability while the inherent nature of GA is global exploration. So, combining GA and PSO would highly improve the algorithm performance [32]. There are few studies using GA and PSO hybrid algorithms for VRPTW. Only Xu *et al*. [33] proposed a GA-PSO hybrid algorithm which is based on real coding and using GA to optimize the population after the PSO operation. It is still a challenging task to design an appropriate hybrid structure tightly coupled GA and PSO to improve the overall performance of the algorithm [34].

In VRPTW, most heuristics have considered hierarchical objectives in which the primary objective is to minimize the number of vehicles and the secondary objective is to minimize the total travel distance. However, few studies considered the travel distance as the primary objective. Alvarenga *et al*. [35] proposed a two-stage method based on genetic algorithm and set partitioning formulation (CGH) which uses islands of GA to handle the whole problem in the first stage and reduced the problem to 30% for further evolution in the second stage. All routes in the best individual from each island are included in the set of routes R. The set partitioning problem (SPP) model is solved over the subset R to obtain the best combination of routes in a unique solution. Subsequently, Labadi [36] proposed a memetic algorithm using the giant tour for chromosome encoding, which is a genetic algorithm hybridized with several local search algorithms. Ursani *et al*. [37] proposed a local genetic algorithm(LGA) based on localized optimization framework which solved the smaller sub-problems decomposed from the whole problem first and then constructed the global problem from the sub-problems. The construction of overlapping sub-problems and the de-optimization of global solutions make LOF be a powerful decomposition scheme. Compared with CGH, LGA has achieved better results. Yu *et al*. [38] proposed a tabu search and ant colony optimization (ACO-Tabu) hybrid algorithm using ant colony optimization with local search to obtain an approximately optimal solution and using tabu search to maintain population diversity and explore the new solution. Similarly, Zhang *et al*. [39] proposed a hybrid algorithm by combining the tabu search and the artificial bee colony algorithm (Tabu-ABC). After the artificial bee colony algorithm has generated an initial feasible solution, tabu search is applied to generate a high-quality solution rapidly. The experimental results of Tabu-ABC are better than ACO-Tabu. Xu *et al*. [33] combined GA and PSO in the form of real code. The main body of the algorithm is PSO, and the crossover operator of GA is used to improve the population diversity. The experiment results of the algorithm are very competitive. Some multi-objective algorithms have also achieved excellent results in minimizing the total travel distance. Tan *et al*. [19] proposed a hybrid multi-objective evolutionary algorithm (HMOEA) that incorporates specialized genetic operators and local search methods featured with variable-length chromosome representation and Pareto fitness ranking. Qi *et al*. [43] presented a multi-objective evolutionary algorithm based on decomposition (MOEA/D) framework for solving the VRPTW. It decomposes the whole problem into a set of scalar sub-problems with uniformly distributed aggregation weight vectors and minimizes these scalar sub-problems simultaneously by evolving a population of solutions. This multi-objective algorithm achieved outstanding results in minimizing the travel distance. The analysis of the results of some multi-objective algorithms shows that there is an indeterminate relationship between these two objectives, sometimes positively correlating sometimes conflicting [23], [44]. Positively correlating means that the total travel distance is increased as the number of vehicles is increased, while conflicting means that the total travel distance is reduced as the number of vehicles is increased. This relationship is unknown until the problem has been solved and varies according to different test datasets [19]. However, there is a lack of research on using this relationship to improve the performance of algorithms.

The travel distance is the most critical objective [15], [45] in cases where it has a higher economic impact (e.g., fuel consumption and environmental requirements) and where the mission time is required accurate (e.g., transporting perishable goods). As shown in Fig.1, different from the conventional VRPTW with the primary objective of minimizing the number of vehicles, the solution space of this problem can be divided into three classes of sub-solution spaces: the solution space with the same number of vehicles, the solution space with the same customers allocation and the single solution. The number of vehicles, i.e., the number of routes, has a strong correlation with the total travel distance. When the number of vehicles reaches some specific value, the total travel distance will get the minimum value. In the solutions space with a same number of vehicles, there are many different types of customers allocations. Customers allocation refers to which customers are included in each route regardless of the sequence of customers. Each kind of customer allocation contains several different solutions.

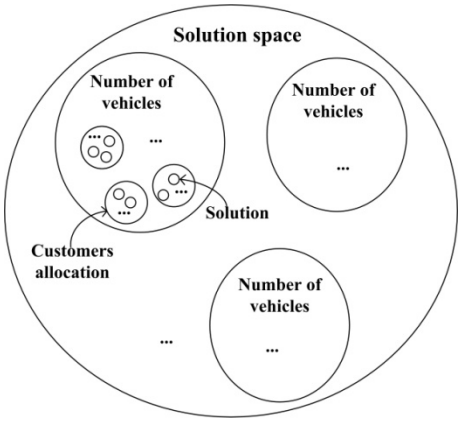


FIGURE 1. Sub-solution spaces.

The primary objective of this paper is to minimize the total travel distance. Therefore, we introduced two algorithms with different search characteristics: evolutionary scatter search (ESS) and route-segment based discrete Particle swarm optimization (PSO). The ESS is a hybridization of scatter search with evolution operators. Two new designed evolution operators: ''route+/−'' and route-exchange based GA are used as the ''solution combination method'' in the scatter search framework. ''Route+/−'' operator is good at finding the ''specific'' number of vehicles, while GA focuses on exploring different customers allocations. The PSO is transformed into discrete form by setting the route-segment as the ''velocity of particles'' and can find better solutions rapidly due to its strong local search capability and fast convergence speed. Based on the concept of ''Memories and Guidance,'' ESS and PSO are combined by a cascade learning architecture that each particle in PSO learns from the ''exemplary solutions'' stored in the reference set which is maintained by ESS. The selection method adjusts selection criteria according to the relationship between the number of vehicles and the travel distance to guide the algorithm searching in more potential direction. All the algorithms are designed based on ''ruin and recreate'' rules while randomness is introduced into the heuristic insertion methods.

The main contributions of this paper can be summarized as follows:

(1) proposed a new discrete PSO based on route-segment to solve the VRPTW.

(2) designed a cascade learning architecture that tightly coupled scatter search and PSO to efficiently integrate the search characteristics of different algorithms and improve the overall performance of the algorithm.

(3) set the criteria of the selection method according to the relationship between the number of vehicles and total travel distance to guide the search direction of the algorithm.

(4) introduced a new solution representation called ''auxiliary code'' to maintain the diversity of the population.

The remainder of this paper is organized as follows. In Section II, we describe the mathematical model of VRPTW. In section III, ESS-PSO is proposed with the entire design process of the algorithm. Experiments with the Solomon benchmark is performed to analyze the impact of different operators on the performance of the algorithm and compared with other excellent algorithms in Section IV. Finally, Section V gives an overall conclusion.

II. MATHEMATICAL MODELS FOR VRPTW

VRPTW is a complex combinatorial optimization problem. In the problem, some customers are waiting to be served, and each customer has its time window and demand. A fleet of identical vehicles should visit all the customers and meet their needs while satisfying the time window constraints. Each vehicle can only serve one customer, and each customer can only be served by one vehicle, a total load of a vehicle cannot exceed its capacity. A solution of the VRPTW is a collection of routes containing an ordered queue of customers, in which a vehicle departs from the depot, visits these customers in sequence and return to the depot. The objective of the designed algorithm is to minimize the total travel distance of all the vehicles. The mathematical model is as follows:

The VRPTW can be defined as a directed complete graph $G(V, A)$ with a node set $V = \{c_0, c_1, \ldots, c_n\}$ and an arc set $A = \{\langle c_i, c_j\rangle): i \neq j, \mathrm{c_i}, \mathrm{c_j} \in V\}$. c_0 represents the depot and $c_i (i = 1, 2, \ldots, \mathrm{n})$ represents the customer. Each node is associated with a demand quantity q_i, a service time s_i and a time window$[e_i, l_i]$ (To the node c_0: $q_0 = 0$, $s_0 = 0$, $e_i = 0$). Each arc $\langle c_i, c_j\rangle$ is associated with a travel time t_{ij} between nodes c_i and c_j. The travel time is represented by the Euclidean distance $d_{ij} (d_{ij} = d_{ji})$. If a vehicle arrives at customer c_i earlier than e_i, it should wait until e_i to serve the customer. And if the vehicle arrives after l_i , it cannot serve c_i. When the vehicle serves the customer c_i , it has to spend at c_i for a time interval at least s_i for service. There are K vehicles with same capacity Q. All the vehicles can depart from the depot after the earliest time e_0 and return to the depot before the latest time l_0.

Objective function:

$$\min TD = \sum_{k=1}^{K}\sum_{i=0}^{N}\sum_{j=0}^{N} t_{ij}x_{ij}^{k} \tag{1}$$

Subject to:

$$x_{ij}^{k} = \begin{cases} 1 & \text{if vehicle } k \text{ travels from } i \text{ to } j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$\sum_{j=1}^{N} x_{0j}^{k} = \sum_{i=1}^{N} x_{i0}^{k} = 1 \quad (\forall k = 1, 2, \ldots, K) \tag{3}$$

$$\sum_{j=0}^{N} x_{ij}^{k} = \sum_{j=0}^{N} x_{ji}^{k} \leq 1 \quad (i \neq j, \forall i = 1, 2, \ldots, N; \forall k = 1, 2, \ldots, K) \tag{4}$$

$$\sum_{k=1}^{K}\sum_{i=0}^{N} x_{ij}^{k} = 1 \quad (i \neq j, \forall j = 1, 2, \ldots, n) \tag{5}$$

$$\sum_{k=1}^{K}\sum_{j=0}^{N} x_{ij}^{k} = 1 \quad (i \neq j, \forall i = 1, 2, \ldots, n) \tag{6}$$

$$\sum_{i=0}^{N} q_i \sum_{j=0}^{N} x_{ij}^{k} \leq Q \quad (i \neq j; \forall k = 1, 2, \ldots, K) \tag{7}$$

$$t_i^{k} + s_i + t_{ij} - t_j^{k} \leq (1 - x_{ij}^{k}) \cdot M \quad (i \neq j, \forall i, j = 0, 1, \ldots, N; \forall k = 1, 2, \ldots, K) \tag{8}$$

$$e_j \sum_{j=0} x_{ij}^{k} \leq t_j^{k} \leq l_j \sum_{j=0} x_{ij}^{k} \quad (i \neq j, \forall i = 0, 1, \ldots, N; \forall k = 1, 2, \ldots, K) \tag{9}$$

$$x_{ij}^{k} \in \{0, 1\} \quad (i \neq j, \forall i, j = 0, 1, \ldots, N; \forall k = 1, 2, \ldots, K) \tag{10}$$

Equation (1) is the objective function of the problem which is to minimize the total travel distance *TD*. Equation (2) defines the decision variable. Equation (3) represents each vehicle departs from the depot and return to the depart at last. Equation (4) is the flow conservation constraints of a node. Equation (5) and (6) ensure that each customer can only be served by one vehicle exactly once. Equation (7) represents that the total demands of customers served by a vehicle cannot exceed its capacity. Equation (8) and(9) define the time window constraint, where t_i is the time the vehicle k arrives at customer i. M is a large constant. Equation (10) imposes binary conditions on the decision variable.

III. THE PROPOSED ALGORITHM

The hybrid algorithm ESS-PSO works in a cascade learning architecture using scatter search with evolution operators to do diversification search, while PSO performs intensification search by learning from excellent ''gene'' of the exemplary solutions in the reference set. The reference set stores the preferred solution observed throughout the evolutionary history, and it is the ''bridge'' that connects the two algorithms. By learning from the good solution in the reference set, it improves the learning capability of the particles which is rigidly constrained in the conventional PSO that each particle is limited to learn from the global best solution and the best solution of its neighbors. In turn, the reference set is updated not only by the scatter search but also by the PSO, which helps scatter search generate better solutions.

The pseudo-code of the whole algorithm is shown in Fig.2. Firstly, the initial population is generated by the diversification generation method of ESS with two improved PFIH insertions: customer-random PFIH and hardest-first PFIH. The initial population is also the initial particle swarm of PSO. Then, the reference set is built by the reference set update method of ESS according to the auxiliary code.

After the initialization of population and reference set, this algorithm enters the main loop. The subset generation method of ESS is employed to generate one solution subsets for the ''route+/−'' operator and for the GA operator. The solution combination method of ESS which contains these two operators is performed over the solutions subsets to obtain a population of new solutions. The reference set is updated by the reference set update method. Then, the route-segment based discrete PSO is performed. The solutions which are also called particles in the particle swarm of PSO search the reference set to find their guidance particles. According to the difference of auxiliary code and objective function value from their guidance particles, the particles are evolved by the new velocity and position updating rules or the local path relinking method. Therefore, the new particle swarm is obtained, and the reference set is updated with these new solutions by the reference set update method. The main loop of ESS-PSO will be repeated until the stopping criterion is reached. The algorithm is described in more detail in the following subsections.

A. EVOLUTIONARY SCATTER SEARCH

Scatter search is a population-based meta-heuristic algorithm that combines preserved solutions from a reference set to create new improved solutions. The reference set stores the preferred solution with high quality and high diversity. Compared with Evolution algorithm (EA), scatter search is a structured strategy, it clearly stated where the improvement method can be applied and avoids using random components such as crossover or mutation operators. However, Nebro *et al*. [42] have proved that applying stochastic operators in the scatter search leads the algorithm to be more robust and accurate. Therefore, two randomized evolution operators: ''route+/−'' and route-exchange based GA, are introduced into scatter search as ''solution combination methods''.

The evolutionary scatter search is based on the scatter search template. The template defines five methods: diversification generation method, improvement method, reference set update method, subset generation method, and solution combination method. In ESS, we did not use any local search procedure, so the improvement method is excluded from the algorithm. However, the designed PSO can be seen as an ''improvement method'' but with its own population. Notes that, the initial population generated by the diversification

1. **Require:** RS_1,RS_2,RS>0 and RS= RS_1+RS_2
/*Initialization*/
2. Generate an initial population of size RS using **diversification generation method** of ESS:
 2.1 Generate 1 solution by **hardest-first PFIH**
 2.2 Generate RS-1 solutions by **customer-random PFIH**
3. Build the **reference set** using **reference set update method** of ESS
/*Main loop*/
4. **for** n=1:N do,
5. Determine the criteria of the **k-tournament selection method**:
 5.1 **if** $|Vd| \leq Vs$
 -Select "the number of vehicles" as criteria
 5.2 **else**
 -Select "the total travel distance" as criteria
6. Perform **subset generation method** of ESS:
 6.1 **for** i=1:RS do,
 6.2 Generate an one solution subset:
 -Select the current solution S_i as the first solution
 6.3 Generate a two solutions subset:
 -Select the current solution S_i as the first solution
 -Use **k-tournament selection method** to select a solution from the **reference set** as the second solution
 end for (i)
7. Perform **solution combination method** of ESS:
 7.1 Perform **GA** operator with the two solutions subsets:
 for j=1:RS do,
 -Use **route-exchange crossover** to generate two child solutions
 -Use **customer-extracted mutation** for the two child solutions to generate two new mutated child solutions
 -Move all the four child solutions into the population of new solutions
 end for (j)
 7.2 Perform **route+/-** operator with the one solution subsets:
 for j=1:RS do,
 -Set the number of vehicles of the current solution S_j as V_j
 if V_j< VB
 -Use **route+** operator to generate a new solution
 else if V_j> VB
 -Use **route-** operator to generate a new solution
 else
 -Randomly select one of the **route+/-** to generate a new solution.
 -Move the new solution into the population of new solutions
 end for (j)
8. Update the **reference set** using **reference set update method** of ESS
9. Perform **route-segment based PSO**
 9.1 **for** i=1: RS do,
 9.2 Compare the **auxiliary code** between the current particle S_i and all solutions in the **reference set**
 9.3 **if** a solution S' that has the same **auxiliary code** with S_i is found
 9.4 Compare the objective function value between S_i and S'
 9.5 **if** S_i is better
 -Set S' as the guidance particle
 -Use **local path relinking method** to evolve S_i
 9.6 **else**
 -Use **k-tournament selection method** to select a solution from the **reference set** as the guidance particle
 -Use **velocity updating rules** and **position updating rules** to evolve S_i
 9.7 **else**
 -Randomly select a solution from the **reference set** as the guidance particle
 -Use **velocity updating rules** and **position updating rules** to evolve S_i
10. Update the **reference set** using **reference set update method** of ESS
11. **end for** (n)

FIGURE 2. ***(Continued.)*** **Pseudo-code of the whole algorithm.**

generation method of ESS is also the initial particles swarm of PSO, and the diversification generation method would not enter the main loop of the ESS-PSO algorithm.

The design process of the algorithm is detailed below.

1) SOLUTION REPRESENTATION

The representation of the solution is the basis of the evolutionary algorithms and has a significant impact on the implementation and performance of the algorithms. For VRPTW, each solution is composed of several routes, and each route contains the sequence of the customer to be served. The algorithm runs directly on this phenotype of the solution. In particular, we designed a genotype code called "auxiliary code" to represent the customers allocation of the solution. The customers allocation refers to which customers are included in each route regardless of the sequence of customers.

As shown in Fig.3, solution 1 and solution 2 are different, but each route contains the same customer in these

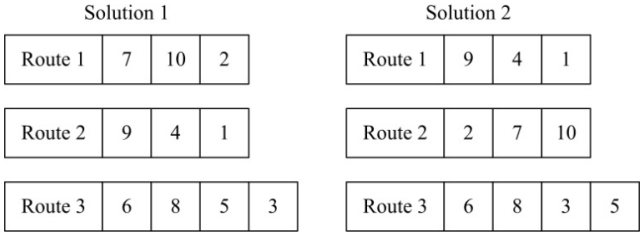


FIGURE 3. **Two similar solution.**

two solutions, and the difference is only the sequence of the customers. These two solutions are very similar because of their same customers allocation. When there are a lot of similar solutions in the population, the diversity of the population will decline, and the algorithm is easily trapped in local optima. Therefore, using auxiliary code to distinguish individuals can maintain the diversity of populations and remove the redundant solutions. Also, the auxiliary code can reduce the calculation scale and help the operators run efficiently.

The auxiliary code consists of N integers, N is the total quantity of customers, customers are placed in the order of its number $1, 2, \ldots N$, Let $A = [a_1, a_2, \ldots, a_i, \ldots, a_N]$ represents an auxiliary code of a solution. There is a route $R = [c_1, c_2, \ldots, c_j, \ldots, c_M]$, c_j is the ''customer number'' at the jth position in route R. Sort route R by customer number from small to large, $R_{sort} = [c_{\min}, \ldots, c_{prev}, c_i, \ldots, c_{\max}]$. The auxiliary code of c_i is $a_i = c_{prev}$, if c_i is at the head of the route $a_i = c_{\max}$. Take the above two solutions as an example, the auxiliary code of solution 1 is $A_1 = [9, 10, 8, 1, 3, 5, 2, 6, 4, 7]$, the auxiliary code of solution 2 is $A_2 = [9, 10, 8, 1, 3, 5, 2, 6, 4, 7]$, these two solutions' auxiliary code are identical.

2) DIVERSIFICATION GENERATION METHOD

This method is used to generate the initial solutions with diversity. In this algorithm, solutions are distinguished by the auxiliary code, so the population is inherently diverse. Since ESS-PSO is base on ''Ruin and recreate,'' the insertion methods that generate the initial population is also used for all operators.

Push forward insertion heuristics (PFIH) is a classic heuristic insertion method used in VRPTW [3]. The traditional PFIH selects the customer farthest from the depot or the customer with the earliest start time as a seed customer to construct a new route and then inserts the optimal customer into its optimal position of the new route with the least insertion cost. The seed customer is the first customer of the new route. We introduced two improved PFIH: customer-random PFIH and hardest-first PFIH as the insertion method. Customer-random PFIH randomly selects an unrouted customer as the next customer to be inserted. It greatly increases the diversity of solutions by introducing randomness. Hardest-first PFIH selects the unrouted customer with the shortest time window interval as the next customer to be inserted. By firstly handling the customers that are difficult to insert, the convergence and optimization capabilities of the method are improved. The insertion criteria used in these two methods is the cost of the distance from inserting the selected customer between two consecutive customers on the same route.

3) REFERENCE SET UPDATE METHOD

This method maintains and updates the reference set. The reference set is a population of solutions with high-quality and high-diversity. The traditional scatter search method generally uses two subsets to store solutions according to their quality and diversity respectively. In ESS, auxiliary code is used to distinguish each solution to maintain the population diversity. Also, we construct two subsets but with different evaluation criteria and scale. The subset $refset_1$ is the set of the best solutions with minimum total travel distance found in the searching process, while subset $refset_2$ store the solutions with the least number of vehicles. By comparing the solutions of the two subsets, we can clearly understand the difference number of vehicles between the solution with the minimum travel distance and the solution with the least vehicles, which will guide the algorithm to search in the direction of more promising ''number of vehicles''. The size of $refset_1$ is *much larger than* $refset_2$ because the main role of the latter is to help identify the relationship between the number of vehicles and the total travel distance.

4) SUBSET GENERATION METHOD

This method generates subsets from the reference set which are used in the combination method to create new solutions. The most usual strategy considers all pairwise combinations of solutions in the reference set. In ESS, we generate different subsets for different operators. For the ''route+/−'' operator, each subset consists of only one solution, each solution in the reference set is one subset. For the GA operator, the subset consists of two solutions, solutions in the reference set must be the first solution in one subset once and only once, the second solution is selected by the designed selection method.

The selection method is based on the k-tournament method with two criteria: the number of vehicles and the total travel distance. The selection method dynamically changes the criteria by the relationship between the number of vehicles and the total travel distance to guide the search direction of the algorithm. Let the difference value of the number of vehicles between the best solution in the $refset_2$(the solution with the least vehicles) and the global best solution (the solution with the least total travel distance) be $|Vd|$, define the steering threshold as Vs. When $|Vd| \leq Vs$, select the number of vehicles as criteria; when $|Vd| > Vs$, select the total travel distance as criteria. It is because the relationship between the number of vehicles and the travel distance is more likely to be positively correlating when the number of vehicles between the solution with the least vehicles and the global best solution is similar. In this case, solutions with fewer vehicles are preferred, and the algorithm is guided to search in the direction of fewer vehicles. On the contrary, if there is a significant difference in the number of vehicles, the relationship is nearly conflicting. So, it is unnecessary to consider the influence of the number of vehicles and directly select solution according to the travel distance. The selection method is also used in the PSO.

5) SOLUTION COMBINATION METHOD

This method transform a given subset of solutions into new solutions. In ESS, we introduced two evolution operators to search the solution space in different directions according to

their characteristics. The first one is route-exchange based GA which is good at exploring the promising type of "customers allocation." The second one is "route+/−" operator, it focuses on finding the appropriate number of vehicles to reduce the total travel distance.

(1) Due to the strong constraints in VRPTW, the traditional genetic operators will produce a lot of infeasible solutions, so the insertion-based crossover operator and mutation operator are used in GA. The insertion-based crossover operator is called route-exchange crossover used in [19]. The solutions in the crossover operation would share their routes to generate two new solutions. Different from Tan *et al*. [19], the routes for exchanging are randomly selected, and the unrouted customers after exchanging are inserted into the partial solution by the two designed insertion methods. The process of crossover is shown in Fig.4.

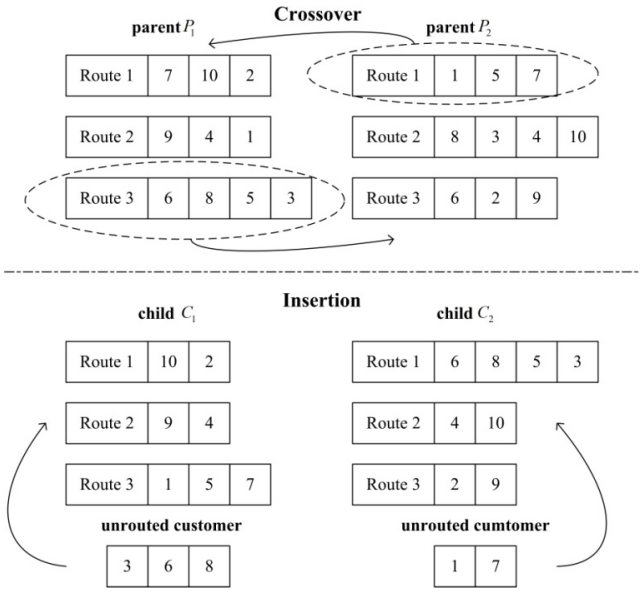


FIGURE 4. The process of route-exchange crossover.

The insertion-based mutation operator is called customer-extracted mutation which extracts some customers randomly and reinserts them back to the solution by the designed insertion methods. Each customer in the solution determines whether it is extracted according to the mutation probability. The process of mutation is shown in Fig. 5

(2) "Route+/−" is also an insertion-based operator. It increases or decreases the number of vehicles in the selected solution according to the relationship between the number of vehicles and the total travel distance.

First, compare the number of vehicles of the currently selected solution *VI* with the number of vehicles of the global best solution *VB*. When $VI < VB$, it indicates that the solution with few vehicles also has a good objective function value and there is potential to continue searching in the direction of few vehicles, use the "route −" operator for the selected solution. On the contrary, when $VI > VB$, use the "route +" operator for the selected solution to continue searching in the

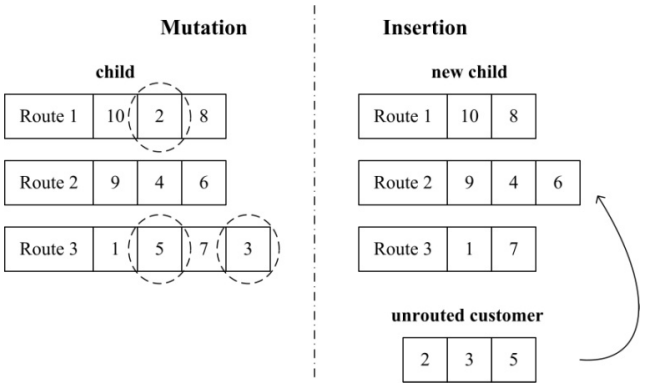


FIGURE 5. The process of customer-extracted mutation.

direction of more vehicles. When $VI = VB$, it is difficult to determine which direction is more potential, so randomly choose an operator to keep searching.

"route −" operator randomly deletes *Rp* routes from the selected individual and adds the customer of the deleted routes in the unrouted customer set. $R_p = rand[1, Rp_{\max}]$, where *rand* denotes a random selection operator, $Rp_{\max}$ denotes the upper limit of the number of routes can be deleted. Then, $Rp - 1$ routes are created using the customer in the unrouted customer set, and each route has only one seed customer. The heuristic insertion method is randomly used to insert the unrouted customer to the partial solution. "route +" operator creates $Rp + 1$ routes after deleting *Rp* routes, the other parts are the same as the "route −" operator. The process of the two operators is shown in Fig. 6.

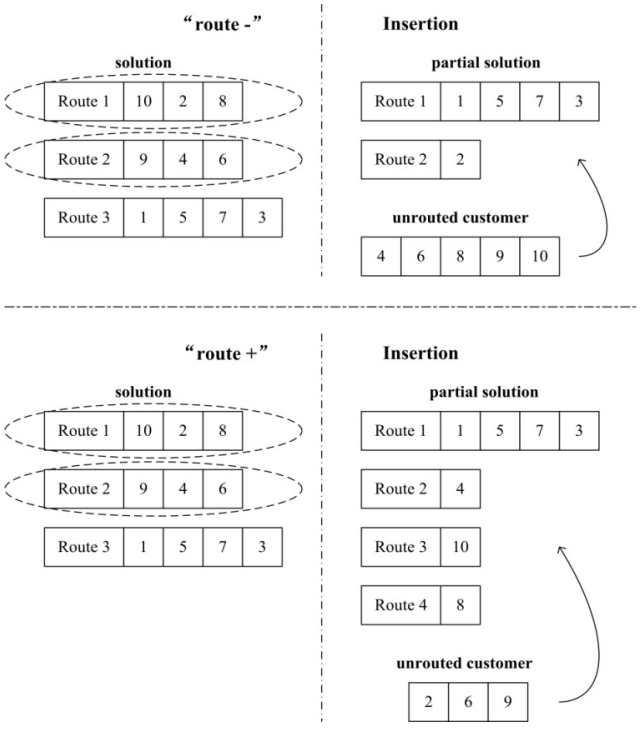


FIGURE 6. The process of "route −" and "route +" operator.

The workflow of ESS is shown in Fig.7.

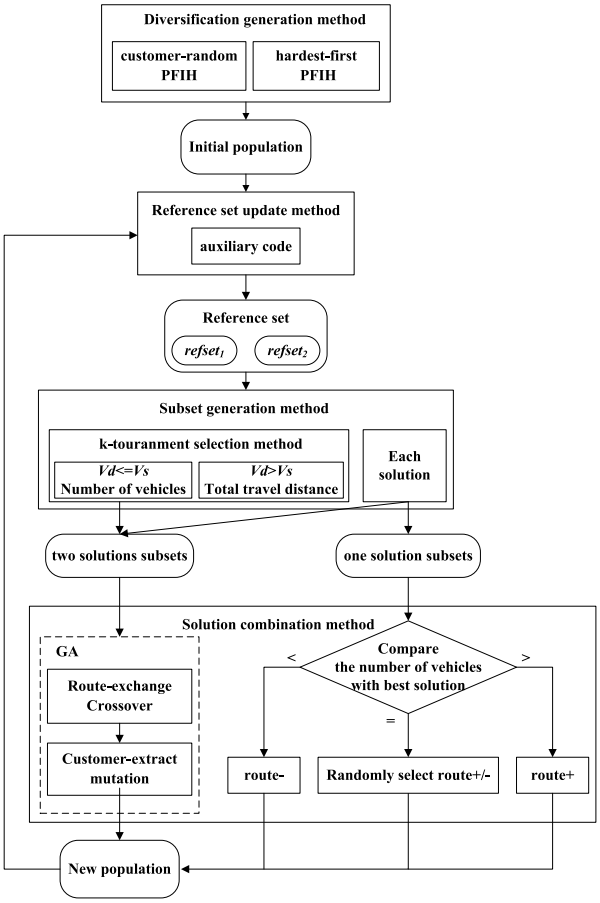


FIGURE 7. Workflow of ESS.

B. DISCRETE PARTICLE SWARM OPTIMIZATION

PSO is a kind of population-based iterative stochastic optimization algorithm. The core of PSO is the learning strategy which involves two issues: which solution to choose as exemplars, and how to use them [50]. The standard PSO keeps learning from the previous best solution of each particle and of its neighbors, which restricted particles search neighborhoods and make them hardly jump out of local optimum. So, we proposed a route-segment based discrete PSO algorithm, in which particles(solutions) learns from the reference set maintained by the ESS. Yin *et al*. [49] have proven that the PSO with particles learning from the reference set performs better in the term of both solution quality and robustness than the standard version.

In this algorithm, we designed two learning strategies depending on the auxiliary code of each particle solution. If there is a solution in the reference set that has the same auxiliary code but a better objective function value, the current particle learns from it by the ''local path relinking'' method. Otherwise, the selection method is used to select a solution with a better objective function value from the reference set to be the guidance particle. The current particle learns from it by the new designed position update rules. By learning from the reference set updated by ESS, the designed PSO can search in the vicinity of the more promising ''number of vehicles'' and ''customers allocation'' to find better solutions.

The conventional PSO is used for continuous optimizations and needs to be modified when solving discrete problems such as VRPTW. In route-segment based discrete PSO, each particle learns from only one ''teacher'' which is called guidance particle. The particle's position is directly represented by the routes with a sequence of customers. The velocity is consist of a set of route segments with possibilities. Route segment is a partial route with some customers which are appeared together in a specific route both in the guidance particle and current particle. In the position update rules, the route segments will be selected by their possibility to construct a new position of the particle. The position update rules are defined as an operator using insertion method.

1) PARTICLE REPRESENTATION

The position of a particle is represented by:

$$X_i(t) = [X_i^1, X_i^2, \ldots X_i^k \ldots, X_i^D] \tag{11}$$

$$X_i^k = [c_{i,1}^k, c_{i,2}^k, \ldots c_{i,n}^k \ldots, c_{i,N_c^k}^k], \quad k \in \{1, D\} \tag{12}$$

$X_i(t)$ is the position of the i th particle at the t th iteration, it contains D routes. The kth dimension X_i^k is the kth route of the ith particle, $c_{i,n}^k$ is the customer which is at the nth order in the kth route, N_c^k is the total quantity of customers in the kth route.

2) VELOCITY UPDATING

The velocity of the particle is obtained by comparing the routes between the guidance particle and current particle. It consists of the route segment and the selection probability of the route segment:

$$V_i(t) = [V_i^1, V_i^2, \ldots V_i^k \ldots, V_i^{D_v}] \tag{13}$$

$$V_i^k = \{(R_x, p_x) | R_x \in \bigcup_{i \in D} (X_d^k - X_i)\} \tag{14}$$

$$p_x = \frac{S_x}{\sum\limits_{i \in N_R^k} S_i} \tag{15}$$

$V_i(t)$ is the velocity of the ith particle at the tth iteration, D_v is the total dimension of the velocity and also the number of routes in guidance particle. V_i^k is the kth dimension of the velocity, R_x is the xth route segment in V_i^k, $p_x \in [0, 1]$ is the selection probability of R_x. p_x denotes the possibility of R_x to be selected in position updating step. X_d^k is the kth dimension (i.e., kth route) of the guidance particle, $\bigcup_{i \in D} (X_d^k - X_i)$ is the set of all the route segments generated by the kth route of the guidance particle. N_R^k is the number of route segments generated by the kth route of the guidance particle. S_x is the score of the route segment R_x,it is obtained from the total number of customers in R_x and the number of customers in R_x which were both consecutive in the being compared routes.

Route segments and the selection probability are obtained from the following steps: Select the kth route of the guidance particle and compare with all the routes of the current particle. Select the customers who both appeared in the being compared routes of guidance particle and current particle to be a route segment. Then select the next route in the guidance particle and repeat the above process until all the routes have been selected. Note that the segment with only one customer is discarded because one customer does not have any ''neighborhood information'' with other customers which cannot afford helpful guidance. Each route segment gets a score based on the number of customers it contains and the consecutive customer segment is rewarded because it contains stronger route information. For example, the selected route in the guidance particle is [1-4-8-5-9], the current particle contains three routes [4-7-2], [8-1-10-3] and [5-9-6]. Then, three route segments [4], [1-8]/[8-1] and [5-9] are obtained. The first segment is discarded, the second segment has a score of 2, the third segment has a score of 3 because of two consecutive customers contained. So, the segment set contains two segments, the selection probability p_1 of segment [1-8]/[8-1] is 0.4, the selection probability p_2 of segment [5-9] is 0.6.

3) POSITION UPDATING

The position of the particles is updated after the velocity of the particles is obtained:

$$X_i(t+1) = Insert(V_i(t), pW) \tag{16}$$

$X_i(t+1)$ is the updated position of the particle, pW represents the selection probability of the customers' sequence in the route segment, the function ''$Insert()$'' represents inserting the customers outside the route segment to all selected route segments. Firstly, route-segment is selected according to its selection probability p_x. Then, a random number between [0,1] is generated and compared with pW. When the random number is larger than pW, the customers' sequence of the selected route segment is the same as them in the guidance particle; otherwise, it is the same as them in the current particle. Then, randomly select an insertion method to insert the customers outside the route-segment into all the selected route-segment. Finally, a new position of the particle is generated. The process of position updating is shown in Fig.8.

4) LOCAL PATH RELINKING

When the auxiliary code is the same as its guidance particle, i.e. their customers allocations are identical, the current particle will get less evolution on using the designed position updating rules. Therefore, we introduced a local path relinking strategy to search for more solutions in this situation.

Based on the path relinking method [51], the local path relinking works on a single route each time. The detail process is as follows: select one route from the current particle to be the ''starting solution,'' the route with the same customers in guidance particle is set as ''target solution.'' The method of simple swapping of two nodes is used to transform the ''starting solution'' to ''target solution.'' If a new better route is found during the transform, the current route is replaced with the new one and the next route in the current particle is selected to be ''starting solution''; otherwise, the next route is selected until the ''starting solution'' is completely transformed to the ''target solution.''

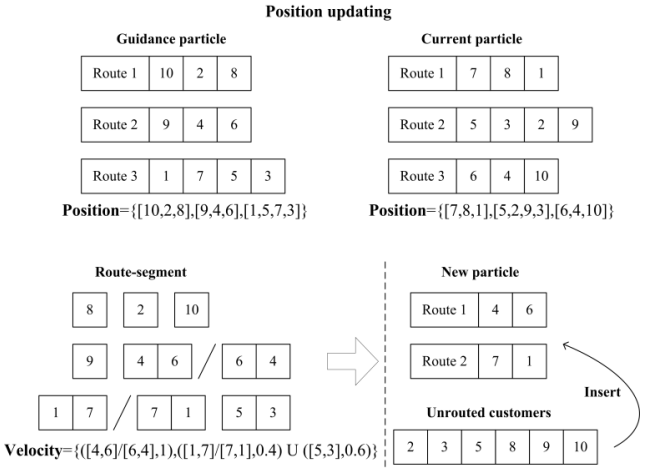


FIGURE 8. The process of position updating.

The workflow of the designed PSO is shown in Fig.9.

C. THE ANALYSIS OF THE ALGORITHM

Since the GA, route+/− and PSO is designed based on the ''ruin and recreate'' rules, the time complexity of all the algorithm mainly depends on the improved PFIH methods. In addition, the reference set update method also plays a major role. Assume N is the number of customers (the dimension of the problem), M is the size of the reference set and Q is the size of particle swarm in PSO. The time complexity of the insertion methods to construct a new solution is $O(N^3)$, and the time complexity of the insertion methods to insert the unrouted customers to the partial solution is $O(N^2)$. Therefore, the time complexity of population initialization is $O(QN^3)$, the time complexity of GA and Route+/− are $O(MN^2)$ and the time complexity of PSO is $O(QN^2)$. The time complexity of updating the reference set for one new solution is $O(M^2)$. So, the time complexity of updating the reference set after initialization and performing PSO is $O(QM^2)$, and the time complexity of updating the reference set after performing GA and route+/− is $O(5M^3)$. Assume that the algorithm runs T iterations, the time complexity of ESS-PSO can be calculated as

$$\begin{aligned} &O((QN^3+QM^2)+((MN^2+MN^2+5M^3)+(QN^2+QM^2))T) \\ &\quad = O(QN^3 + MN^2T + M^3T + QN^2T + QM^2T) \end{aligned}$$

IV. EXPERIMENT RESULTS

Solomon benchmark with 100 customers is used to verify the quality of the algorithm. The benchmark contains six classes of problems: C1, C2, R1, R2, RC1 and RC2, a total of 56 test instances. The depot and 100 customers are distributed in a Cartesian coordinate space of 100x100, and

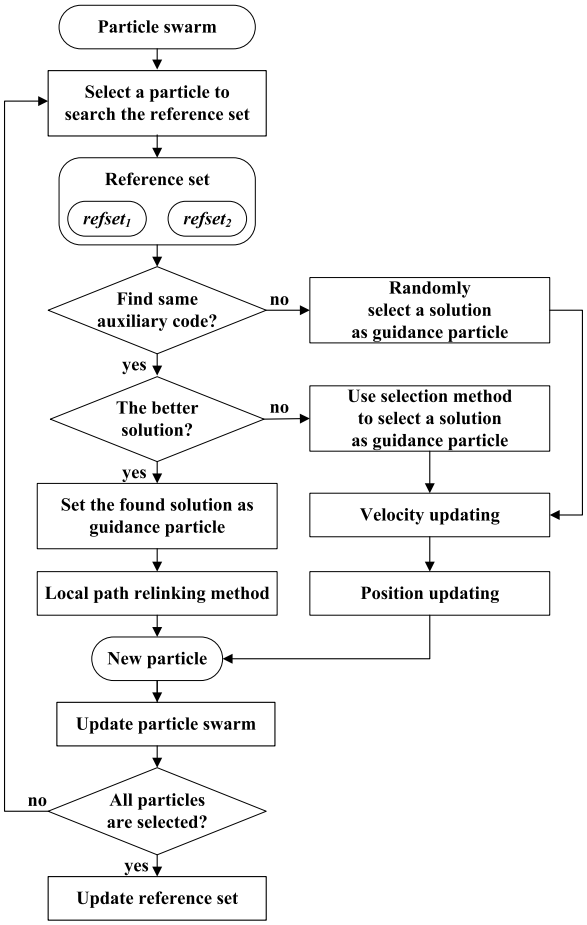


FIGURE 9. Workflow of the designed PSO.

the distance between customers is calculated using a simple Euclidean distance while assuming that 1 unit distance equals 1 unit travel time. "R" represents the random distribution of customers, "C" represents customer located in clusters and "RC" contains a mix of randomly distributed an clustered customers. Customers in the category "1" have a narrow time window and each vehicle can only serve a small number of customers with less capacity. Customers in the category "2" have a wide time window, and each vehicle can serve more customers with larger capacity. The proposed algorithm is programmed in C ++ and runs on a 64-bit win7 system using an Intel Core i7 2.9 GHz CPU with 8GB of memory.

A. EFFECTIVENESS OF THE OPERATORS

In order to understand the influence on the performance of using the relationship between the number of vehicles and the total travel distance to guide search direction, we compare three different scales of the subset *refset2* in the reference set. The size of *refset2* RS_2 is set to: 1/5 of the size of *refset1* RS_1, 1/10 of RS_1 and 0. Steering threshold $Vs = 1$. The size of the whole reference set RS is set to 50, $RS = RS_1 + RS_2$. The selection method in algorithm without *refset2* directly select the total travel distance as the criteria. The rest of the algorithm and the parameter in the experiment is the same between the three algorithms. As all instances in the classes of C1 and C2 have positively correlating objectives, there are many instances in R1, R2, RC1 and RC2 that are having conflicting objectives [19]. We select several instances from categories of "R" and "RC," and run 10 times for each instance. The obtained results are shown in Table 1. In the table, "minNV" represents the best-known results of minimizing the number of vehicles, "minTD" represents the best-known results of minimizing the total travel distance, "TD" represents the results of average distance, "NV" represents the average number of vehicles, "T" represents the average operation time of the proposed algorithm in seconds.

The two instances from category "R1" have positively correlating objectives. In these instances, algorithms "1/5" and "1/10" outperform the algorithm "0" with less total travel distance and fewer vehicles. The same result is also obtained in "RC204" but with the same number of vehicles between the three algorithms. It indicates that keeping searching in the direction of the fewer number of vehicles brings less total travel distance in these instances. In category "RC1", the two instances which are more difficult to solve also has positively correlated objectives. Algorithm "1/10" gets the best result but the difference between the results of algorithms "1/5" and "0" are not obvious. The number

TABLE 1. Comparison of the selection method.

	minNV		minTD		"1/5"			"1/10"			"0"		
Dataset	TD	NV	TD	NV	TD	NV	T	TD	NV	T	TD	NV	T
R103	1292.68	13	1213.62	14	1223.62	14.20	37	1223.84	14.20	36	1227.32	14.8	37
R108	960.88	9	947.55	10	962.45	10.20	38	964.97	10.40	37	967.30	10.50	37
RC101	1696.94	14	1623.58	15	1663.01	15.80	33	1657.72	15.80	34	1663.05	16.00	33
RC106	1424.73	11	1371.69	13	1403.32	13.20	32	1398.80	13.00	34	1405.38	13.30	34
R202	1191.70	3	1034.35	8	1050.76	5.80	44	1050.34	5.80	44	1049.69	5.90	43
R206	906.14	3	879.89	5	892.95	4.70	54	891.13	4.70	56	891.66	4.70	56
RC201	1406.91	4	1134.91	6	1285.54	8.30	37	1285.17	8.10	36	1285.31	8.20	36
RC204	798.41	3	786.38	4	792.71	4.00	74	794.00	4.00	76	795.71	4.00	74

"minNV" represents the best-known results of minimizing the number of vehicles, "minTD" represents the best-known results of minimizing the total travel distance, "1/5" represents RS_2=1/5·RS_1, "1/10" represents RS_2=1/10·RS_1, "0" represents RS_2=0, the "TD" represents the results of average distance, "NV" represents the average number of vehicles, "T" represents the average operation time of the proposed algorithm in seconds.

TABLE 2. Comparison of different operators.

	(GA,PSO)			(GA, route+/-)			(PSO, route+/-)			(GA, PSO, route+/-)		
Dataset	**TD**	**NV**	**T**	**TD**	**NV**	**T**	**TD**	**NV**	**T**	**TD**	**NV**	**T**
R103	1224.28	14.30	30	1239.88	14.40	33	1231.28	14.60	14	1223.84	14.20	36
R108	967.43	10.80	32	994.05	10.10	35	983.72	10.40	17	964.97	10.40	37
RC101	1664.38	15.80	29	1666.61	16.10	32	1681.17	16.40	13	1657.72	15.80	34
RC106	1401.83	13.30	28	1436.28	13.10	31	1414.37	13.30	13	1398.80	13.00	34
R202	1063.95	4.70	34	1054.02	5.50	40	1063.17	5.70	19	1050.34	5.80	44
R206	901.24	4.00	43	901.02	4.60	50	902.77	4.50	24	891.13	4.70	56
RC201	1337.48	5.60	29	1302.02	8.00	32	1297.78	8.40	15	1285.17	8.10	36
RC204	798.90	3.90	61	797.96	4.00	75	805.28	4.10	36	794.00	4.00	76

"TD" represents the results of average distance, "NV" represents the average number of vehicles, "T" represents the average operation time of the proposed algorithm in seconds.

TABLE 3. Comparison of population size.

	R103		RC101		R202		RC201	
	TD	**T**	**TD**	**T**	**TD**	**T**	**TD**	**T**
(20,20)	1232.07	14	1667.54	13	1066.35	16	1296.11	14
(20,50)	1225.25	33	1666.60	32	1058.16	38	1287.14	32
(20,100)	1225.35	65	1660.56	70	1049.10	74	1284.28	66
(50,20)	1226.16	18	1675.63	18	1068.01	22	1299.66	18
(50,50)	1223.84	36	1657.72	34	1050.34	44	1285.17	36
(50,100)	1222.24	70	1660.90	66	1046.87	80	1286.74	66
(100,20)	1227.64	24	1666.98	22	1056.57	30	1300.92	24
(100,50)	1223.66	42	1665.04	41	1051.23	52	1286.09	44
(100,100)	1222.22	77	1658.62	76	1047.33	90	1281.06	77

"TD" represents the results of average distance, "T" represents the average operation time of the proposed algorithm in seconds.

of vehicles obtained by algorithm ''1/10'' and ''1/5'' are both slightly less than algorithm ''0.'' It is because searching in potential direction gives the algorithm more opportunity to jump out of local optimum although sometimes it did not work. In ''R202,'' ''R206,'' and ''RC201'' which have conflicting objectives, the results of the three algorithms are not significantly different and algorithm ''0'' is slightly better. It proves that because of the small proportion of *refset2*, it does not weaken too much to the searching ability of the algorithm when the instance has conflicting objectives. The three algorithms have almost the same runtime in all instances. Although the absolute difference is not significant, this improvement is necessary for finding the optimal solution. To balance the performance in different instances, we select $RS_2 = 1/10 \cdot RS_1$ as the parameter of the algorithm.

Moreover, we perform some experiments to analyze the impact of different operators on the performance of the algorithm. The operators are employed in pairs, we use (*GA*, *PSO*, *route+/−*) to present which operators are employed. For example, (GA,PSO) means GA and PSO are employed. We select the same instances used in the above experiments and run 10 times for each instance. Results are shown in Table 2, and ''TD'' represents the results of average distance, ''NV'' represents the average number of vehicles, ''T'' represents the average operation time of the proposed algorithm in seconds.

As shown in Table 2, these three operators have different search characteristics and the designed hybrid structure which combined all the operators has achieved obvious dominant results. The results of (GA, route+/−) have shown that the designed GA which is good at exploring more ''customer allocations'' has a strong global search ability. The convergence time of (GA, route+/−) is almost the same as (GA, PSO, route+/−), which indicates that GA has always been exploring the solution space. The runtimes of (PSO, route+/−) are very short, but its results are excellent which demonstrates the powerful local search ability and fast convergence speed of PSO. The (GA, PSO) has achieved almost the same results as (GA, PSO, route+/−) in the instances with positively correlating objectives. It is very effective to hybrid these two algorithms together. The ''route+/−'' which is focus on finding the ''appropriate number of vehicles'' has little effect on the instances with positively correlating objectives, but significantly improves the performance of algorithms in the instance with conflicting objectives. When these three operators are working together, GA and ''route+/−'' lead the algorithm towards high-quality solution space and PSO causes more precise search in these areas.

B. PARAMETER SETTINGS

After the above experimental analysis, some parameters in the algorithm are set as follows: steering threshold $Vs = 1$, $RS_2 = 1/10 \cdot RS_1$, k-tournament method $k = 3$, mutation probability $Pm = 0.1$.

The size of the reference set RS is an important factor in both the computational requirements and solution quality of a scatter search implementation. The size of the particle swarm in PSO PS is also critical. Several experiments are performed to examine the effects of these two parameters. The value for

TABLE 4. Comparison with the best-known result.

Best-known				ESS-PSO						
Dataset	TD	NV	Ref	BTD	NV	MTD	NV	Gap	Std	T
C101	828.94	10	[11]	**828.94**	10	828.94	10	0.00%	0	32
C102	828.94	10	[11]	**828.94**	10	828.94	10	0.00%	0	28
C103	828.06	10	[11]	**828.06**	10	828.06	10	0.00%	0	30
C104	824.78	10	[11]	**824.78**	10	824.78	10	0.00%	0	33
C105	828.94	10	[11]	**828.94**	10	828.94	10	0.00%	0	31
C106	828.94	10	[11]	**828.94**	10	828.94	10	0.00%	0	28
C107	828.94	10	[11]	**828.94**	10	828.94	10	0.00%	0	27
C108	828.94	10	[11]	**828.94**	10	828.94	10	0.00%	0	29
C109	828.94	10	[11]	**828.94**	10	828.94	10	0.00%	0	30
C201	591.56	3	[11]	**591.56**	3	591.56	3	0.00%	0	36
C202	591.56	3	[11]	**591.56**	3	591.56	3	0.00%	0	43
C203	591.17	3	[11]	**591.17**	3	591.17	3	0.00%	0	57
C204	590.60	3	[11]	**590.60**	3	592.47	3	0.00%	3.12	71
C205	588.88	3	[11]	**588.88**	3	588.88	3	0.00%	0	42
C206	588.49	3	[11]	**588.49**	3	588.49	3	0.00%	0	50
C207	588.29	3	[11]	**588.29**	3	588.29	3	0.00%	0	50
C208	588.32	3	[11]	**588.32**	3	588.32	3	0.00%	0	59
R101	1613.59	18	[19]	1642.88	20	1650.11	20	1.82%	7.35	41
R102	1454.68	18	[19]	1472.92	18	1478.03	18	1.25%	6.14	41
R103	1213.62	14	[11]	1213.73	14	1222.2	14	0.01%	4.16	38
R104	974.24	10	[19]	976.61	11	994.87	11	0.24%	13.05	38
R105	1360.78	15	[46]	**1360.76**	15	1365.98	15.23	0.00%	5.8	38
R106	1240.47	13	[46]	**1239.37**	13	1250.16	13	-0.09%	10.13	41
R107	1073.34	11	[46]	**1073.34**	11	1082.16	11	0.00%	6.38	40
R108	947.55	10	[46]	950.59	10	973.57	11	0.32%	10.48	46
R109	1151.84	13	[46]	**1151.84**	13	1170.23	13	0.00%	11.31	40
R110	1072.41	12	[46]	1073.46	12	1098.6	12	0.10%	12.41	40
R111	1053.50	12	[46]	**1053.50**	12	1063.01	12	0.00%	7.1	40
R112	953.63	10	[11]	**953.62**	10	979.18	11	0.00%	12.33	40
R201	1144.48	9	[35]	1152.63	8	1170.79	7.07	0.71%	10.04	41
R202	1034.35	8	[46]	1036.30	6	1046.53	5.7	0.19%	6.22	48
R203	874.87	6	[46]	875.21	6	886.4	5.63	0.04%	8.55	65

TABLE 4. *(Continued.)* **Comparison with the best-known result.**

R204	736.52	4	[46]	737.43	4	751.36	4.03	0.12%	11.71	83
R205	954.16	5	[23]	**954.16**	5	971.6	5.23	0.00%	12.3	53
R206	879.89	5	[46]	884.25	5	894.39	4.67	0.50%	8.35	67
R207	799.86	4	[46]	801.15	4	821.15	3.83	0.16%	13.58	79
R208	705.45	4	[46]	706.86	3	725.99	3	0.20%	5.01	118
R209	859.39	5	[46]	860.11	5	875.71	4.7	0.08%	9.72	73
R210	910.70	5	[46]	912.8	5	930.26	5.3	0.23%	10.9	67
R211	755.96	4	[46]	757.6	4	788.82	3.7	0.22%	24.25	102
RC101	1623.58	15	[11]	1639.75	16	1660.48	16	1.00%	7.2	43
RC102	1461.23	14	[46]	1461.33	14	1486.56	14.8	0.01%	7.4	38
RC103	1261.67	11	[47]	1277.55	12	1314.67	11.9	1.26%	11.98	29
RC104	1135.48	10	[48]	1138.13	10	1155.05	10.07	0.23%	7.81	30
RC105	1518.58	16	[46]	1519.46	15	1566.24	15.83	0.06%	15.72	34
RC106	1371.69	13	[19]	1378.62	13	1406.79	13.1	0.51%	16.83	32
RC107	1212.83	12	[46]	**1212.83**	12	1236.86	12	0.00%	16.25	33
RC108	1117.53	11	[46]	1118.57	11	1140.28	11.03	0.09%	7.16	33
RC201	1134.91	6	[19]	1265.56	9	1285.16	8.1	11.51%	9.05	49
RC202	1095.64	8	[46]	1096.53	8	1111.22	7.43	0.08%	9.33	51
RC203	928.51	5	[46]	**926.82**	5	944.69	4.8	-0.18%	7.28	59
RC204	786.38	4	[46]	**786.38**	4	797.47	4	0.00%	7.16	95
RC205	1157.55	7	[46]	**1157.55**	7	1173.06	7.37	0.00%	12.21	47
RC206	1054.61	7	[46]	1057.83	6	1082.95	5.03	0.31%	9.86	65
RC207	966.08	6	[46]	966.37	6	985.59	5.23	0.03%	9.64	72
RC208	779.31	4	[46]	**779.31**	4	808.06	4.3	0.00%	23.62	105

"TD" represents the best-known result of distance, "NV" represents the number of vehicles, "BTD" represents the best results of distance obtained by the proposed algorithm, "MTD" represents the results of average distance obtained by the proposed algorithm, "Time" represents the average operation time of the proposed algorithm in seconds. "Gap"=(BTD-TD)/BTD is the percentage deviation between the ESS-PSO and the best-known results, "Std" is the standard deviation of the algorithm which is used to measure the stability of the solution

each parameters is set to (20, 50, 100), we use (*RS*, *PS*) to represent each combination. We select four problems from different classes of the dataset and run 10 times for each problem. The results are shown in Table 3, and ''TD'' represents the results of average distance, ''T'' represents the average operation time of the proposed algorithm in seconds.

It can be seen from the experimental results that as the size of the reference set and the size of the population increase, the experimental results are getting better and the runtime is getting longer. However, when the scale of the two collections is too large, the experimental results are hard to go further improvement, but the runtime is still growing. To balance performance and runtime, we choose (50, 50) as the size of the reference set and the size of the particle swarm. The number of iterations for the whole algorithm is set to 50. The algorithm run 30 times on each instance.

C. RESULT ANALYSIS

In Table 4, the proposed algorithm is compared with the best-known results, the data of the best-known results are selected from [43]. In the table, ''TD'' represents the best-known result of distance, ''NV'' represents the number of vehicles, ''BTD'' represents the best results of distance obtained by the proposed algorithm, ''MTD'' represents the results of average distance obtained by the proposed algorithm, ''Time'' represents the average operation time of the pro-

TABLE 5. Comparison with recent published and representative algorithms.

	LGA (2011) [37]		Xu et al. (2015) [33]		M-MOEA/D (2015) [43]		Tabu-ABC (2017) [24]		ESS-PSO	
Dataset	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV
C101	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10
C102	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10
C103	**828.06**	10	**828.06**	10	**828.06**	10	828.07	10	**828.06**	10
C104	**824.78**	10	**824.78**	10	**824.78**	10	**824.78**	10	**824.78**	10
C105	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10
C106	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10
C107	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10
C108	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10
C109	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10	**828.94**	10
C201	**591.56**	3	**591.56**	3	**591.56**	3	**591.56**	3	**591.56**	3
C202	**591.56**	3	**591.56**	3	**591.56**	3	**591.56**	3	**591.56**	3
C203	**591.17**	3	**591.17**	3	**591.17**	3	**591.17**	3	**591.17**	3
C204	**590.60**	3	**590.60**	3	**590.60**	3	594.89	3	**590.60**	3
C205	**588.88**	3	**588.88**	3	**588.88**	3	**588.88**	3	**588.88**	3
C206	**588.49**	3	**588.49**	3	**588.49**	3	**588.49**	3	**588.49**	3
C207	**588.29**	3	**588.29**	3	**588.29**	3	**588.29**	3	**588.29**	3
C208	**588.32**	3	**588.32**	3	**588.32**	3	**588.32**	3	**588.32**	3
R101	1646.9	20	**1642.87**	20	1644.70	20	1643.18	20	**1642.88**	20
R102	1474.28	18	**1472.62**	18	1473.73	18	1460.26	18	1472.92	18
R103	1222.68	15	**1213.62**	14	**1213.62**	14	1217.39	15	1213.73	14
R104	989.53	11	982.01	10	991.91	11	987.61	11	**976.61**	11
R105	1382.78	16	**1360.78**	15	1366.58	15	1363.91	15	**1360.76**	15
R106	1250.11	13	1241.52	13	1249.22	13	1247.90	13	**1239.37**	13
R107	1083.42	12	1076.13	11	1086.22	11	1087.50	12	**1073.34**	11
R108	952.44	10	**948.57**	10	965.52	10	961.85	11	950.59	10
R109	1160.69	13	**1151.84**	13	1155.38	13	1152.99	13	**1151.84**	13
R110	1080.69	12	1080.36	11	1106.03	12	1091.50	12	**1073.46**	12
R111	1057.64	12	**1053.50**	12	1073.82	11	1067.46	12	**1053.50**	12
R112	965	10	960.68	10	981.43	10	973.25	10	**953.62**	10
R201	1156.29	9	**1148.48**	9	1185.79	6	1174.69	6	1152.63	8
R202	1042.25	8	1049.74	7	1049.72	5	1046.10	5	**1036.30**	6

TABLE 5. *(Continued.)* **Comparison with recent published and representative algorithms.**

R203	877.29	6	900.08	5	889.36	5	884.02	5	**875.21**	6
R204	**736.52**	4	772.33	4	743.29	5	750.40	4	737.43	4
R205	960.35	6	970.89	6	954.48	5	960.75	5	**954.16**	5
R206	894.19	6	898.91	5	887.90	4	900.97	4	**884.25**	5
R207	**800.79**	4	814.78	3	809.51	4	809.72	4	801.15	4
R208	**706.86**	3	723.61	3	711.59	3	723.14	5	**706.86**	3
R209	860.63	5	879.53	6	867.47	4	863.12	5	**860.11**	5
R210	948.82	5	932.89	7	920.06	5	927.54	5	**912.80**	5
R211	762.23	5	808.56	4	767.10	4	763.22	4	**757.60**	4
RC101	1660.55	16	**1623.58**	15	1646.65	16	1646.17	16	1639.75	16
RC102	1494.92	15	1466.84	14	1484.48	15	1481.61	14	**1461.33**	14
RC103	1276.05	12	**1261.67**	11	1274.85	11	1280.76	12	1277.55	12
RC104	1151.63	10	**1135.48**	10	1145.79	10	1162.03	11	1138.13	10
RC105	1556.21	16	1618.55	16	1528.61	15	1545.30	16	**1519.46**	15
RC106	1402.25	14	**1377.35**	13	1399.17	13	1401.17	14	1378.62	13
RC107	**1212.83**	12	**1212.83**	12	1235.54	12	1235.28	12	**1212.83**	12
RC108	1133.25	11	**1117.53**	11	1138.95	11	1136.35	11	1118.57	11
RC201	1281.63	10	1387.55	8	1289.94	7	1271.78	7	**1265.56**	9
RC202	1103.47	8	1148.84	9	1118.66	5	1116.21	6	**1096.53**	8
RC203	942	6	945.96	5	940.55	5	941.81	5	**926.82**	5
RC204	796.12	4	798.41	3	792.98	4	801.87	4	**786.38**	4
RC205	1168.89	8	1161.81	7	1187.48	6	1165.82	7	**1157.55**	7
RC206	1060.52	7	1059.89	7	1089.14	5	1072.85	5	**1057.83**	6
RC207	970.97	7	976.40	7	987.88	5	977.11	5	**966.37**	6
RC208	782.7	5	795.39	5	807.83	4	792.33	5	**779.31**	4

"TD" represents the best-known result of distance, "NV" represents the number of vehicles.

posed algorithm in seconds. ''Gap'' = (BTD-TD)/BTD is the percentage deviation between the ESS-PSO and the best-known results, which is used to measure the quality of the algorithm. ''Std'' is the standard deviation of the algorithm which is used to measure the stability of the solution.

The results show that the proposed algorithm has obtained 2 new best solutions and reached 27 best solutions. There is no percentage deviation in the C1 and C2 classes. The average percentage deviation of R1, R2, RC1, RC2, and the total dataset are 0.30%, 0.22%, 0.40%, 1.47%, and 0.38% respectively and most of the percentage deviation are below 0.50%. It shows that the optimization performance of the algorithm is excellent. There is no standard deviation in the C1 class, only C204 in the C2 class has 3.12. The average standard deviation of R1, R2, RC1, RC2, and the total dataset are 8.89, 10.97, 11.29, 11.02, and 7.30 respectively and most of the standard deviation are below 10. It shows that the algorithm is stable and has good robustness. The average runtime of C1, C2, R1, R2, RC1, RC2, and the total dataset are 29.78, 51, 41.17, 72.36, 38.50, 69, and 50.46 in seconds respectively and most of the runtimes are below 60. It shows the algorithm is highly competitive in terms of convergence speed. It is worth mentioning that in C1 and C2 classes the algorithm can find the optimal solution in less than 10 seconds in

TABLE 6. The Wilcoxon signed-ranks non-parametric test.

Dataset	ESS-PSO vs LGA		ESS-PSO vs Xu *et al.*		ESS-PSO vs M-MOEA/D		ESS-PSO vs Tabu-ABC	
C1	1		1		1		0.317	
C2	1		1		1		0.317	
R1	0.002	+	0.139		0.003	+	0.015	+
R2	0.017	+	0.004	+	0.003	+	0.003	+
RC1	0.028	+	0.612		0.017	+	0.012	+
RC2	0.012	+	0.012	+	0.012	+	0.012	+

The symbol "+" represents the null hypothesis is rejected and ESS-PSO gets statistically superior performance on the corresponding instance.

TABLE 7. Average comparison.

	C1	C2	R1	R2	RC1	RC2	Avg
Best-known	828.38	589.86	1175.80	877.78	1337.82	987.87	974.02
HMOEA(2006)[19]	828.74	590.69	1187.35	951.74	1355.37	1068.26	1005.19
CGH(2007)[35]	**828.38**	**589.86**	1196.80	899.90	**1341.70**	1015.90	987.42
ACO-Tabu(2011)[38]	829.01	590.78	1196.96	951.36	1380.55	1095.84	1014.77
LGA(2011)[37]	**828.38**	**589.86**	1188.85	885.78	1360.96	1013.29	984.59
meta-HSA(2015)[40]	838.47	605.41	1207.76	977.19	1381.96	1099.12	1018.32
Xu *et al.*(2015)[33]	**828.38**	**589.86**	1182.04	899.98	1351.73	1034.28	988.33
M-MOEA/D(2015)[43]	**828.38**	**589.86**	1192.35	889.66	1356.76	1026.81	988.16
Tabu-ABC(2017)[39]	**828.38**	590.39	1187.90	891.24	1361.08	1017.47	986.88
ESS-PSO	**828.38**	**589.86**	**1180.22**	**879.86**	1343.28	**1004.54**	**978.54**

most instances except for a few problems. More appropriate termination rules should be designed to improve the running speed further.

D. COMPARISON WITH OTHER METHODS

In Table 5, we selected several more recently published and representative algorithms for complete data comparison. The best results obtained for each instance are highlighted in bold.

The results show that the proposed algorithm achieves 45 of the best results in 56 datasets, with 32 of Xu *et al*., 22 of LGA, 18 of M-MOEA / D and 15 of Tabu-ABC. In the C1 and C2 classes, all algorithms except the Tabu-ABC achieve the best results for all datasets. In the R1 and RC1 classes, the proposed algorithm gets 9 and 3 best results respectively, while Xu *et al*. gets 7 and 6. To other algorithms, only LGA gets 1 in the RC1 class. In the R2 class, the proposed algorithm gets 8 best results, while LGA gets 3 and Xu *et al*. gets 1. The proposed algorithm gets all 8 best results in RC2 class. Also, the difference between the result obtained by the proposed algorithm and the best result is very small even in the dataset that the algorithm does not get the best results. It can be seen that the proposed algorithm has the best comprehensive ability and occupies an absolute superiority in the category ''2'' problem with longer time windows, but it is slightly inferior to Xu *et al*. in the category ''1'' problem with narrow time windows. This result shows that it is very effective to adjust the search direction of the algorithm by analyzing the relationship between the number of vehicles and travel distance. However, due to the lack of a powerful method which is more time-consuming to reduce the number of vehicles, the searching ability of the algorithm in a fewer vehicles direction is weaker when the two objectives have a positive correlation. Also, it is more effective to improve the performance of the algorithm by maintaining the diversity of the population in the category ''2'' problem where the vehicle contains more customers, that is, each auxiliary code represents more different individuals.

The non-parametric Wilcoxon signed-ranks is employed to compare the ESS-PSO with the above four algorithms for each instance (pairwise comparison) to determine whether the differences in performance are statistically significant or not. Table 6 shows the p-value resolved by the Wilcoxon test with the significance level $\alpha = 0.05$. The symbol ''+'' represents the null hypothesis is rejected and ESS-PSO gets statistically superior performance on the corresponding instance. In C1 and C2 classes, ESS-PSO is not significantly different with all the four algorithms. In R1 and RC1 classes, except Xu *et al.*, ESS-PSO outperforms other 3 algorithms. Finally, in R2 and RC2 classes, ESS-PSO outperforms all the four algorithms. The results of

the non-parametric test show the outstanding superiority of the ESS-PSO for solving the ''2'' category problems again.

In Table 7, we select eight state-of-art algorithms for the average comparison. The table lists the average minimum travel distance obtained by each algorithm and the best-known result over the 6 classes of problems. The best results for each class are highlighted in bold.

The results show that the proposed algorithm almost achieved the best results in all classes and is close to the best-known results, but only slightly inferior to the CGH algorithm in the RC1 class. Similar to the previous comparison, it proved that although the algorithm is better at solving the category ''2'' problems, it also performs very well in the category ''1'' problems.

V. CONCLUSION

This paper presents a hybrid algorithm of evolutionary scatter search and PSO to solve VRPTW with the objective of minimizing the total travel distance. In ESS, two algorithms: a GA with route-exchange crossover and customer-extract mutation and a newly designed evolution operator ''route+/−'' are used as solution combination methods. Also, we designed a route-segment based PSO to make the algorithm run efficiently in the discrete space for solving VRPTW. Taking the reference set of scatter search as the core, ESS-PSO combined these three population-based algorithms which have different searching characteristics into a systematic structure to improve the overall performance of the hybrid algorithm. We introduced ''auxiliary code'' to distinguish each solution in the reference set to maintain the population diversity. Inspired by the multi-objective algorithm, the search direction of the algorithm is adjusted by the relationship of the number of vehicles and the total travel distance.

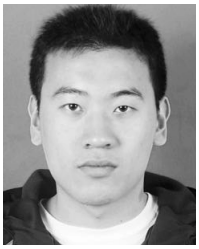
The experimental results show that the proposed algorithm is both effective and efficient, and has obvious advantages in solution quality and running speed compared with other algorithms.

Although the algorithm has achieved excellent results, it did not use any local search algorithms. In the next step, we can add some local search algorithms in the structure to improve the effectiveness and stability of the algorithm further. We would also like to apply the algorithm to other combinatorial optimization problems, especially the various variant problems of VRP.

REFERENCES

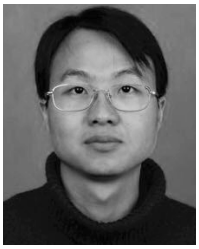
[1] R. A. Russell and W.-C. Chiang, ''Scatter search for the vehicle routing problem with time windows,'' *Eur. J. Oper. Res.*, vol. 169, pp. 606–622, Jan. 2006.
[2] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse, ''The vehicle routing problem: State of the art classification and review,'' *Comput. Ind. Eng.*, vol. 99, pp. 300–313, Sep. 2016.
[3] M. M. Solomon, ''Algorithms for the vehicle routing and scheduling problems with time window constraints,'' *Oper. Res.*, vol. 35, no. 2, pp. 254–265, 1987.
[4] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger, ''Subset-row inequalities applied to the vehicle-routing problem with time windows,'' *Oper. Res.*, vol. 56, no. 2, pp. 497–511, 2008.
[5] D. Pecin, C. Contardo, G. Desaulniers, and E. Uchoa, ''New enhancements for the exact solution of the vehicle routing problem with time windows,'' *Inf. J. Comput.*, vol. 29, no. 3, pp. 489–502, 2017.
[6] O. Bräysy and M. Gendreau, ''Vehicle routing problem with time windows, part II: Metaheuristics,'' *Transp. Sci.*, vol. 39, no. 1, pp. 119–139, Feb. 2005.
[7] M. Schneider, F. Schwahn, and D. Vigo, ''Designing granular solution methods for routing problems with time windows,'' *Eur. J. Oper. Res.*, vol. 263, pp. 493–509, Dec. 2017.
[8] T. Vidal, T. G. Crainic, M. Gendreau, and C. Prins, ''A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows,'' *Comput. Oper. Res.*, vol. 40, no. 1, pp. 475–489, Jan. 2013.
[9] Y. Nagata, O. Bräysy, and W. Dullaert, ''A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows,'' *Comput. Oper. Res.*, vol. 37, pp. 724–737, Apr. 2010.
[10] J. Nalepa and M. Blocho, ''Adaptive memetic algorithm for minimizing distance in the vehicle routing problem with time windows,'' *Soft Comput.*, vol. 20, no. 6, pp. 2309–2327, 2016.
[11] Y. Rochat and E. D. Taillard, ''Probabilistic diversification and intensification in local search for vehicle routing,'' *J. Heuristics*, vol. 1, pp. 147–167, Sep. 1995.
[12] P. Badeau, F. Guertin, M. Gendreau, J.-Y. Potvin, and E. Taillard, ''A parallel tabu search heuristic for the vehicle routing problem with time windows,'' *Transp. Res. C, Emerg. Technol.*, vol. 5, pp. 109–122, Apr. 1997.
[13] J. Li, J. Zhang, C. Jiang, and M. Zhou, ''Composite particle swarm optimizer with historical memory for function optimization,'' *IEEE Trans. Cybern.*, vol. 45, no. 10, pp. 2350–2363, Oct. 2015.
[14] E. E. Zachariadis, C. D. Tarantilis, and C. T. Kiranoudis, ''An adaptive memory methodology for the vehicle routing problem with simultaneous pick-ups and deliveries,'' *Eur. J. Oper. Res.*, vol. 202, pp. 401–411, Apr. 2010.
[15] R. Baños, J. Ortega, C. Gil, A. L. Márquez, and F. de Toro, ''A hybrid meta-heuristic for multi-objective vehicle routing problems with time windows,'' *Comput. Ind. Eng.*, vol. 65, no. 2, pp. 286–296, 2013.
[16] D. Q. Wu, M. Dong, H. Y. Li, and F. Li, ''Vehicle routing problem with time windows using multi-objective co-evolutionary approach,'' *Int. J. Simul. Model.*, vol. 15, no. 4, pp. 742–753, Dec. 2016.
[17] A. L. Bouthillier, T. G. Crainic, and P. Kropf, ''A guided cooperative search for the vehicle routing problem with time windows,'' *IEEE Intell. Syst.*, vol. 20, no. 4, pp. 36–42, Jul. 2005.
[18] D. Barbucha, ''A cooperative population learning algorithm for vehicle routing problem with time windows,'' *Neurocomputing*, vol. 146, pp. 210–229, Dec. 2014.
[19] K. C. Tan, Y. H. Chew, and L. H. Lee, ''A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows,'' *Comput. Optim. Appl.*, vol. 34, pp. 115–151, May 2006.
[20] D. Pisinger and S. Ropke, ''A general heuristic for vehicle routing problems,'' *Comput. Oper. Res.*, vol. 34, no. 8, pp. 2403–2435, Aug. 2007.
[21] J.-F. Cordeau and M. Maischberger, ''A parallel iterated tabu search heuristic for vehicle routing problems,'' *Comput. Oper. Res.*, vol. 39, no. 9, pp. 2033–2050, Sep. 2012.
[22] P. P. Repoussis, C. D. Tarantilis, and G. Ioannou, ''Arc-guided evolutionary algorithm for the vehicle routing problem with time windows,'' *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 624–647, Jun. 2009.
[23] B. Ombuki, B. J. Ross, and F. Hanshar, ''Multi-objective genetic algorithms for vehicle routing problem with time windows,'' *Appl. Intell.*, vol. 24, no. 1, pp. 17–30, 2006.
[24] D. M. Pierre and N. Zakaria, ''Stochastic partially optimized cyclic shift crossover for multi-objective genetic algorithms for the vehicle routing problem with time-windows,'' *Appl. Soft Comput.*, vol. 52, pp. 863–876, Mar. 2017.
[25] J. Luo, X. Li, M.-R. Chen, and H. Liu, ''A novel hybrid shuffled frog leaping algorithm for vehicle routing problem with time windows,'' *Inf. Sci.*, vol. 316, pp. 266–292, Jan. 2015.
[26] L. Tan, F. Lin, and H. Wang, ''Adaptive comprehensive learning bacterial foraging optimization and its application on vehicle routing problem with time windows,'' *Neurocomputing*, vol. 151, pp. 1208–1215, Mar. 2015.
[27] X. Liang, W. Li, Y. Zhang, and M. Zhou, ''An adaptive particle swarm optimization method based on clustering,'' *Soft Comput.*, vol. 19, no. 2, pp. 431–448, Feb. 2015.
[28] E. T. Yassen, M. Ayob, M. Z. A. Nazri, and N. R. Sabar, ''An adaptive hybrid algorithm for vehicle routing problems with time windows,'' *Comput. Ind. Eng.*, vol. 113, pp. 382–391, Nov. 2017.

[29] V. Kachitvichyanukul, "A particle swarm optimization for vehicle routing problem with time windows," *Int. J. Oper. Res.*, vol. 6, no. 4, pp. 519–537, Jan. 2009.
[30] W. Hu, H. Liang, C. Peng, B. Du, and Q. Hu, "A hybrid chaos-particle swarm optimization algorithm for the vehicle routing problem with time window," *Entropy*, vol. 15, no. 4, pp. 1247–1270, 2013.
[31] Y.-J. Gong, J. Zhang, O. Liu, R.-Z. Huang, H. S.-H. Chung, and Y.-H. Shi, "Optimizing the vehicle routing problem with time windows: A discrete particle swarm optimization approach," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 2, pp. 254–267, Mar. 2012.
[32] H. Garg, "A hybrid PSO-GA algorithm for constrained optimization problems," *Appl. Math. Comput.*, vol. 274, pp. 292–305, Feb. 2016.
[33] S. H. Xu, J. P. Liu, F. H. Zhang, L. Wang, and L. J. Sun, "A combination of genetic algorithm and particle swarm optimization for vehicle routing problem with time windows," *Sensors-Basel*, vol. 15, pp. 21033–21053, Aug. 2015.
[34] Y.-J. Gong *et al.*, "Genetic learning particle swarm optimization," *IEEE Trans. Cybern.*, vol. 46, no. 10, pp. 2277–2290, Oct. 2016.
[35] G. B. Alvarenga, G. R. Mateus, and G. de Tomi, "A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows," *Comput. Oper. Res.*, vol. 34, pp. 1561–1584, Jun. 2007.
[36] N. Labadi, C. Prins, and M. Reghioui, "A memetic algorithm for the vehicle routing problem with time windows," *RAIRO-Oper. Res.*, vol. 42, pp. 415–431, Jul. 2008.
[37] Z. Ursani, D. Essam, D. Cornforth, and R. Stocker, "Localized genetic algorithm for vehicle routing problem with time windows," *Appl. Soft Comput.*, vol. 11, pp. 5375–5390, Dec. 2011.
[38] B. Yu, Z. Z. Yang, and B. Z. Yao, "A hybrid algorithm for vehicle routing problem with time windows," *Expert Syst. Appl.*, vol. 38, pp. 435–441, Jan. 2011.
[39] D. Zhang, S. Cai, F. Ye, Y.-W. Si, and T. T. Nguyen, "A hybrid algorithm for a vehicle routing problem with realistic constraints," *Inf. Sci.*, vols. 394–395, pp. 167–182, Jul. 2017.
[40] E. T. Yassen, M. Ayob, M. Z. A. Nazri, and N. R. Sabar, "Meta-harmony search algorithm for the vehicle routing problem with time windows," *Inf. Sci.*, vol. 325, pp. 140–158, Dec. 2015.
[41] W. Dong and M. Zhou, "A supervised learning and control method to improve particle swarm optimization algorithms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 7, pp. 1135–1148, Jul. 2017.
[42] A. J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J. J. Durillo, and A. Beham, "AbYSS: Adapting scatter search to multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 4, pp. 439–457, Aug. 2008.
[43] Y. Qi, Z. Hou, H. Li, J. Huang, and X. Li, "A decomposition based memetic algorithm for multi-objective vehicle routing problem with time windows," *Comput. Oper. Res.*, vol. 62, pp. 61–77, Oct. 2015.
[44] K. Ghoseiri and S. F. Ghannadpour, "Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm," *Appl. Soft Comput.*, vol. 10, pp. 1096–1107, Sep. 2010.
[45] J. de Armas, B. Melián-Batista, J. A. Moreno-Pérez, and J. Brito, "GVNS for a real-world rich vehicle routing problem with time windows," *Eng. Appl. Artif. Intell.*, vol. 42, pp. 45–56, Jun. 2015.
[46] S. Jung and B. R. Moon, "A hybrid genetic algorithm for the vehicle routing problem with time windows," in *Proc. 4th Annu. Conf. Genet. Evol. Comput.*, New York, NY, USA, 2002, pp. 1309–1316.
[47] P. Shaw, "Using constraint programming and local search methods to solve vehicle routing problems," in *Proc. Int. Conf. Princ. Pract. Constraint Program.*, 1998, pp. 417–431.
[48] J.-F. Cordeau, G. Laporte, and A. Mercier, "A unified tabu search heuristic for vehicle routing problems with time windows," *J. Oper. Res. Soc.*, vol. 52, no. 8, pp. 928–936, Aug. 2001.
[49] P. Yin, F. Glover, M. Laguna, and J.-X. Zhu, "Cyber swarm algorithms—Improving particle swarm optimization using adaptive memory strategies," *Eur. J. Oper. Res.*, vol. 201, pp. 377–389, Mar. 2010.
[50] Z. Ren, A. Zhang, C. Wen, and Z. Feng, "A scatter learning particle swarm optimization algorithm for multimodal problems," *IEEE Trans. Cybern.*, vol. 44, no. 7, pp. 1127–1140, Jul. 2014.
[51] Y. Marinakis and M. Marinaki, "A hybrid genetic—Particle swarm optimization algorithm for the vehicle routing problem," *Expert Syst. Appl.*, vol. 37, pp. 1446–1455, Mar. 2010.
[52] Z. Zhang, Y. Sun, H. Xie, Y. Teng, and J. Wang, "GMMA: GPU-based multiobjective memetic algorithms for vehicle routing problem with route balancing," *Appl. Intell.*, vol. 48, pp. 1–16, Jun. 2018.
[53] J. Wang, W. Ren, Z. Zhang, H. Huang, and Y. Zhou, "A hybrid multiobjective memetic algorithm for multiobjective periodic vehicle routing problem with time windows," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published. [Online]. Available: https://ieeexplore.ieee.org/document/8438859
[54] F. Zaman, S. M. Elsayed, T. Ray, and R. A. Sarker, "Configuring two-algorithm-based evolutionary approach for solving dynamic economic dispatch problems," *Eng. Appl. Artif. Intell.*, vol. 53, pp. 105–125, Aug. 2016.
[55] F. Zaman, S. M. Elsayed, T. Ray, and R. A. Sarkerr, "Evolutionary algorithms for finding nash equilibria in electricity markets," *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 536–549, Aug. 2018.



JINGTIAN ZHANG received the B.S. degree from the School of Automation, Huazhong University of Science and Technology, Wuhan, China, in 2010. He is currently pursuing the Ph.D. degree in mechatronic engineering with the Beijing University of Posts and Telecommunications, Beijing, China. His current research interests include intelligent control system of autonomous vehicles and logistics automation.

FUXING YANG received the Ph.D. degree from the School of Mechatronics Engineering, Harbin Institute of Technology, Harbin, China, in 1996. From 1996 to 1999, he was with the National Defense Key Laboratory of Ultra-Precision Processing Technology, Aviation Industry Corporation of China. From 1999 to 2001, he was with the China Academy Of Engineering Physics. Since 2001, he has been with the Beijing University of Posts and Telecommunications, Beijing, China, where he is currently a Professor and the Ph.D. Supervisor with the Department of Logistics Engineering. His research interests include logistics information technology, Internet-of-Things application technology, and advanced manufacturing technology.



XUN WENG received the Ph.D. degree from the School of Mechanical Engineering, University of Science and Technology Beijing, Beijing, China, in 2007. Since 2007, he has been with the Beijing University of Posts and Telecommunications, Beijing, where he is currently an Associate Professor and the Director of the Department of Logistics Engineering. His research interests include logistics equipment and automation, logistics center planning, and logistics center equipment design.